Università degli Studi di Trento
Facoltà di Scienze Matematiche, Fisiche e Naturali
Dipartimento di Ingegneria e Scienza dell'Informazione

Awareness Requirements for Adaptive Systems

Vítor E. Silva Souza, Alexei Lapouchnian[1], William N. Robinson[2], John Mylopoulos

vitorsouza@disi.unitn.it

[1] University of Toronto, Canada / [2] Georgia State University, USA

Photo is Copyright © 2008 Andrea Caranti (http://www.flickr.com/photos/11102419@N00/1476590956/). Used with author explicit permission.

License to use, adapt and distribute

This material is available for any kind of use and can be derived and/or redistributed, as long as it uses an equivalent license and attributes credit to original authors.

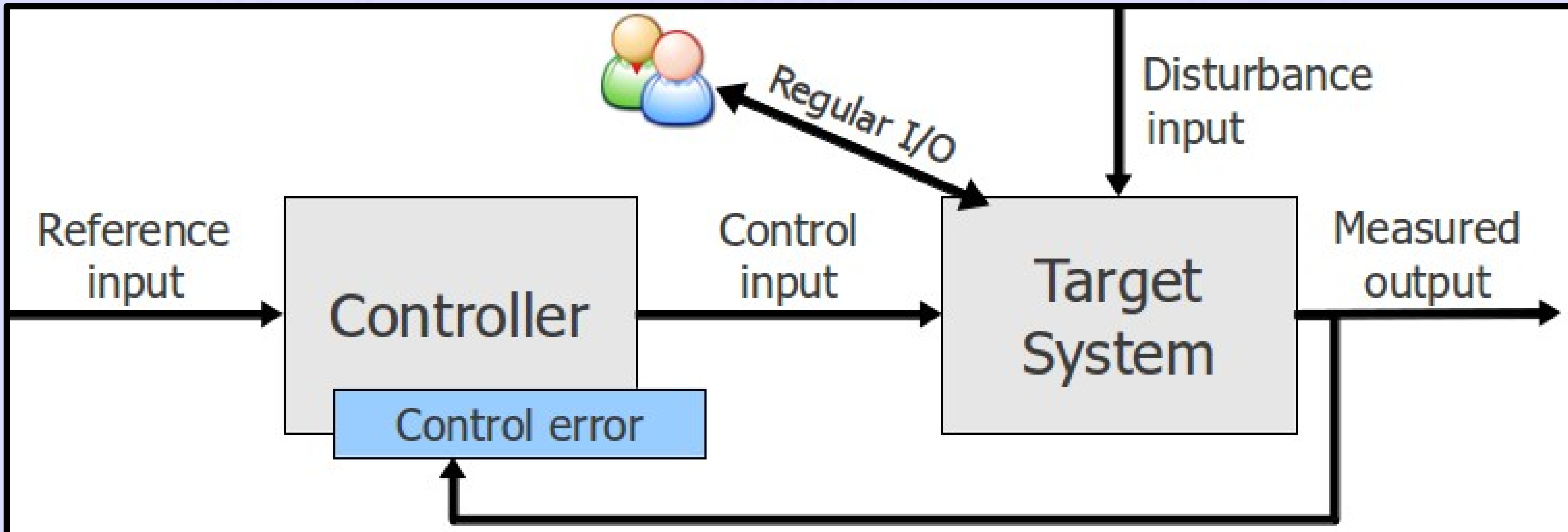
Attribution-Share Alike 3.0 Unported

http://creativecommons.org/licenses/by-sa/3.0/

You are free to copy, distribute, transmit and adapt this work under the following conditions: (a) You must attribute the work in the manner specified by the author or licensor (but not in any way that suggests that they endorse you or your use of the work); (b) If you alter, transform, or build upon this work, you may distribute the resulting work only under the same, similar or a compatible license.

Adaptive Systems

- Growing interest (SEAMS, ICAC, SASO, ...);
- Feedback loop architectures [Andersson et al., 2009; Brun et al., 2009];
  - MAPE feedback loop [Kephart & Chess, 2003].

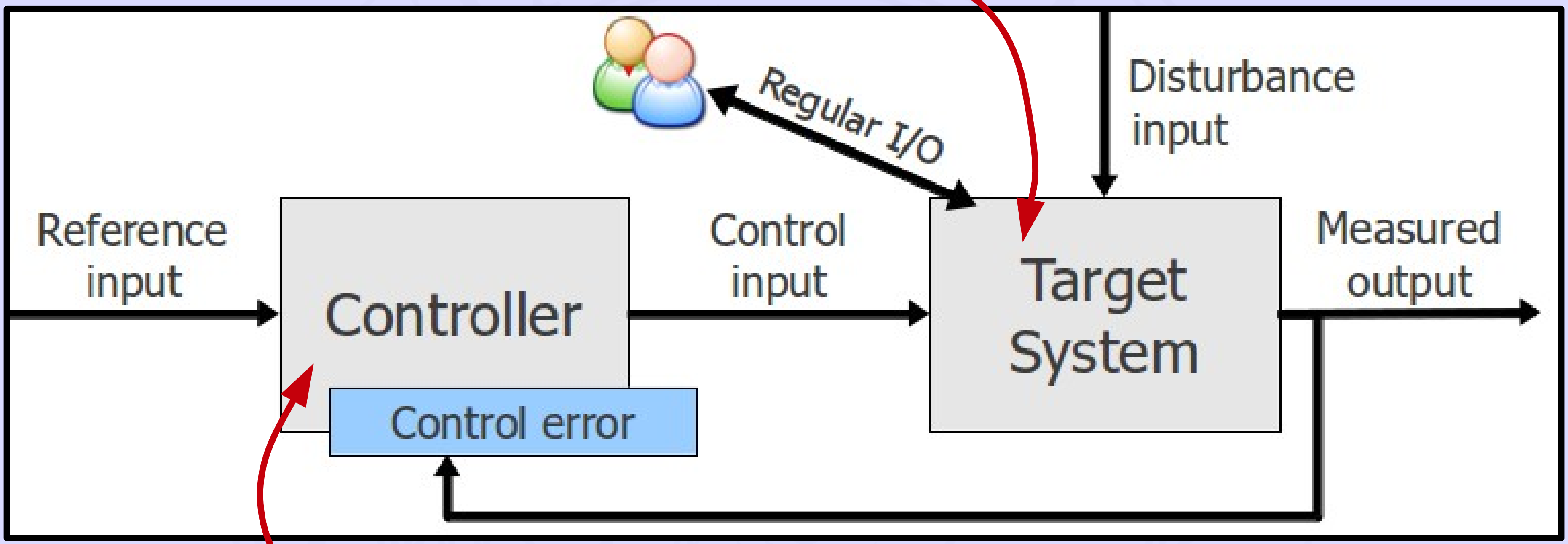


Adapted from [Hellerstein et al., 2004]

CC BY SA

RE-oriented Perspective

We know the requirements for this

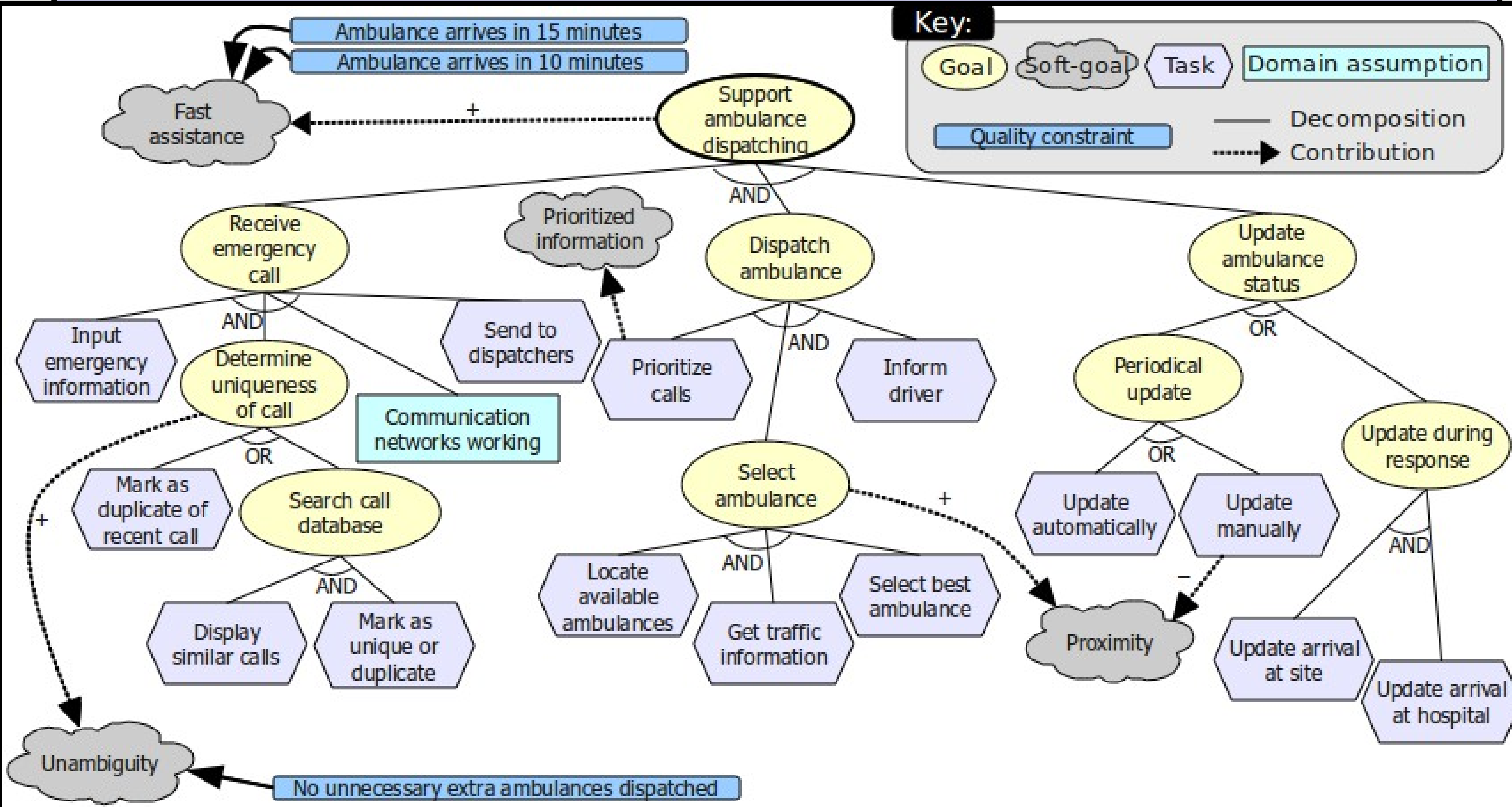


What are the requirements for this?

"... the objective ... is to make some output, say y, behave in a desired way by manipulating some input, say u ..." [Doyle et al., 1990]

CC BY SA

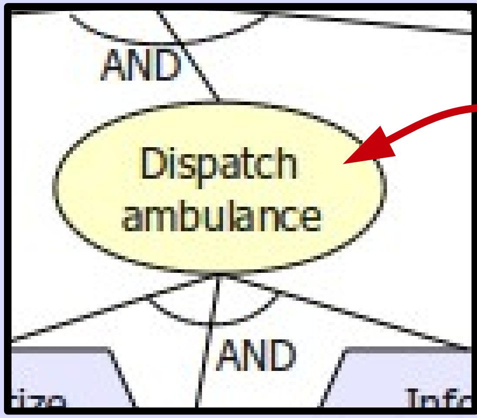
GORE perspective



Based on [Jureta et al., 2008]

cc BY SA

Adaptivity over Requirements



Target system requirement:
R = "Dispatch ambulance to emergency site"

Possible monitoring requirements for the controller:
R' = "Every execution of R succeeds / R never fails"
R'' = "R succeeds 95% of the time over any one-month period"

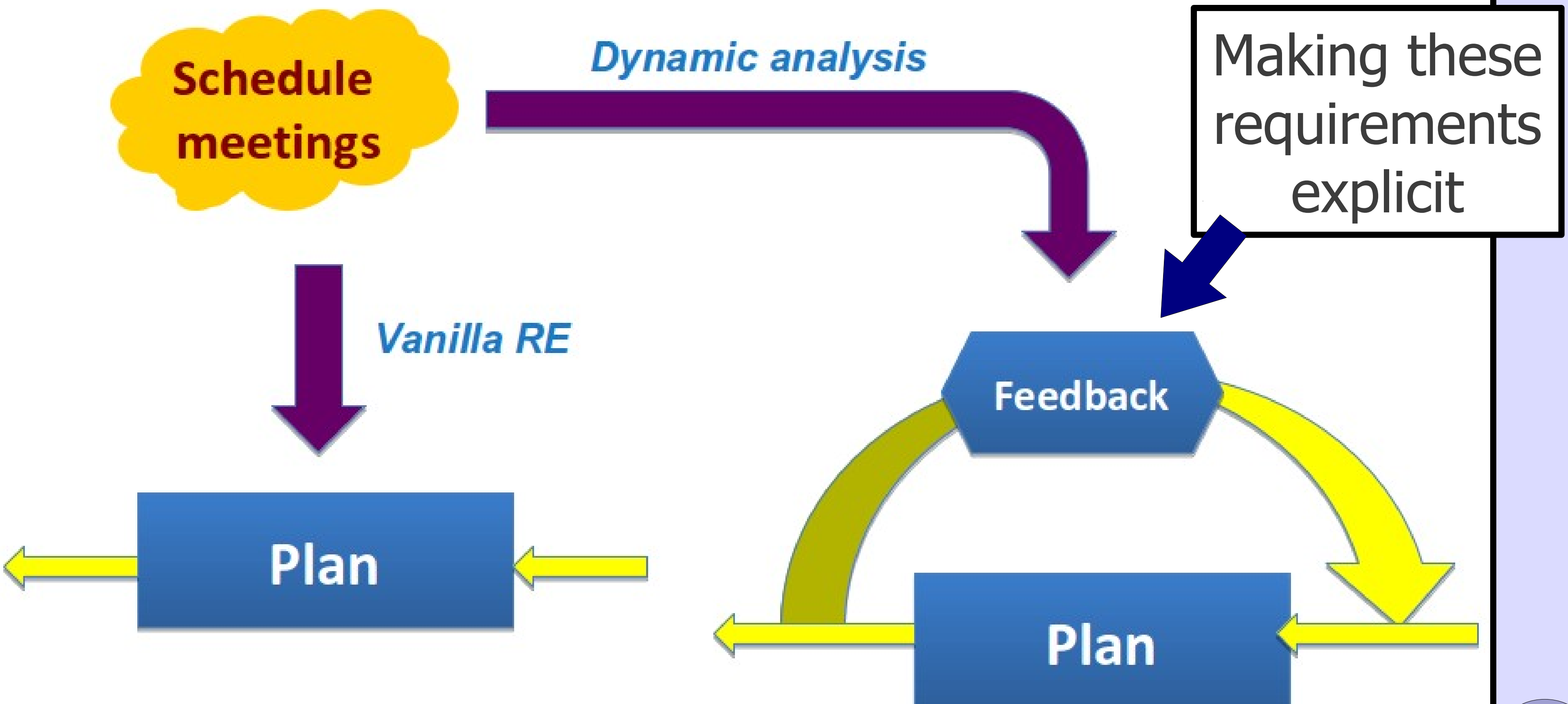
Awareness Requirements

CC BY SA

John Mylopoulos' keynote – SEAMS '10

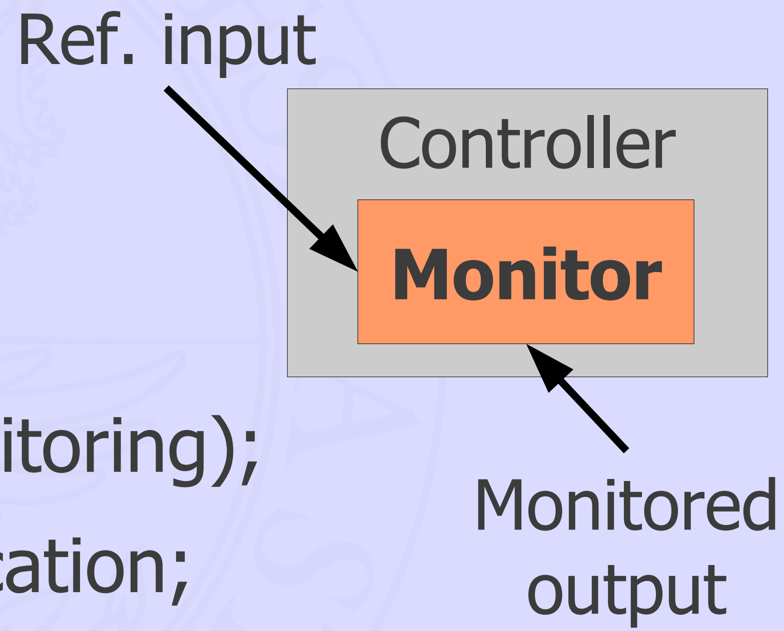
Baseline

- FLEA+KAOS [Feather98], ReqMon [Robinson06], …



Awareness Requirements (AwReqs)

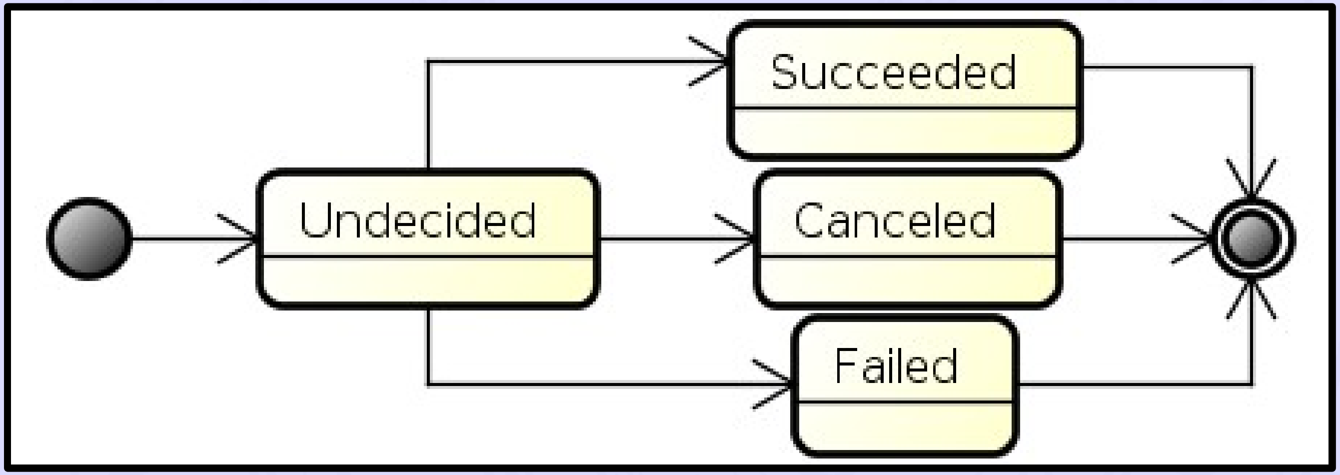
- Requirements that refer to other requirements and their success/failure at runtime;
- Outline:
  - Characterization / elicitation;
  - Formalization;
  - Validation (requirements monitoring);
  - A glimpse on System Identification;
- Agenda for the future:
  - Systematic process for modeling/design of AS;
  - Complete MAPE loop prototype.



CC BY SA

Requirements as run-time objects

- AwReqs refer to the states that instances of requirements can assume at runtime;



cc BY SA

Categories of AwReqs (1)

- **Basic / regular / “never fail”:**
  - *Input emergency information* should never fail;
- **Aggregate (I):**
  - *Communication networks working* should have 99% success rate;
  - *Ambulance arrives in 10 min* should succeed 60% of the time, while *Ambulance arrives in 15 min* should be successful 80%, measured daily;

BY SA

Categories of AwReqs (2)

- **Trend (D):**
  - The success rate of *No unnecessary extra ambulances* for a month should not decrease, compared to the previous month, twice in a row;
- **Delta (P):**
  - *Update arrival at site* should be successfully executed within 10 minutes of *Inform driver*;
  - *Mark as unique or duplicate* should be decided in 5 min.

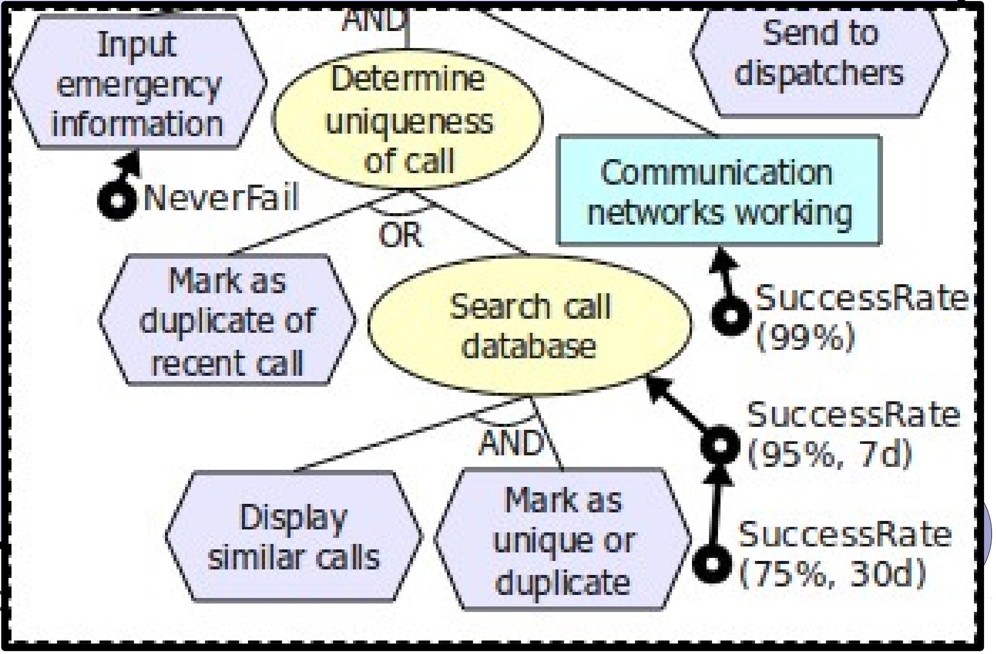
cc BY SA

AwReq patterns*

- NeverFail(T-InputInfo);
- SuccessRate(D-CommNetsWork, 99%);
- @daily SuccessRate(Q-Amb10min, 60%) and SuccessRate(Q-Amb15min, 80%);
- not TrendDecrease(Q-NoExtraAmb, 30d, 2);
- ComparableDelta(T-UpdArrSite, T-InfDriver, time, 10m);
- StateDelta(T-MarkUnique, Undecided, *, 5m).

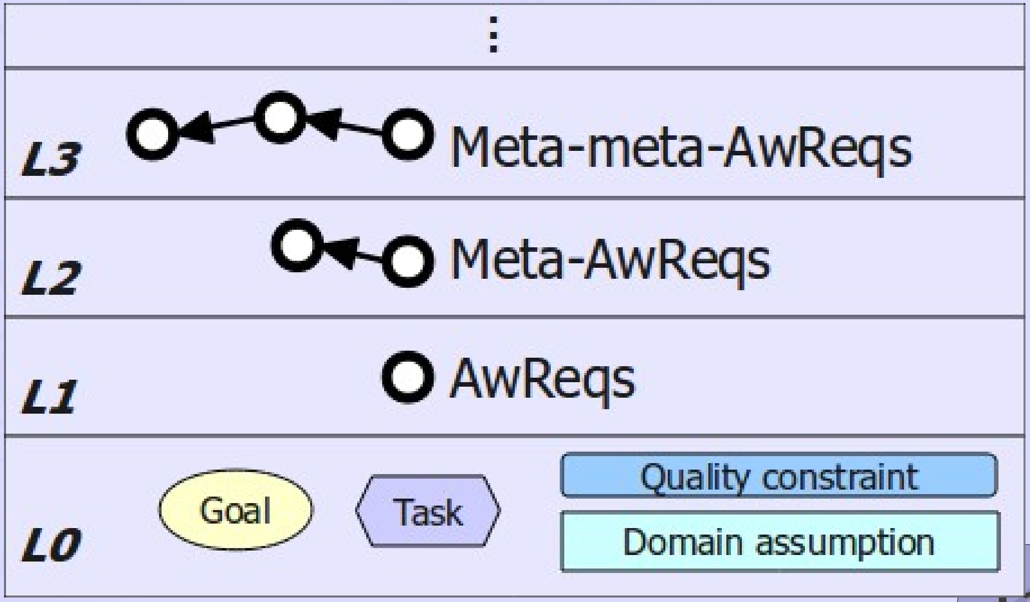
We could use qualitative values

* non exhaustive!



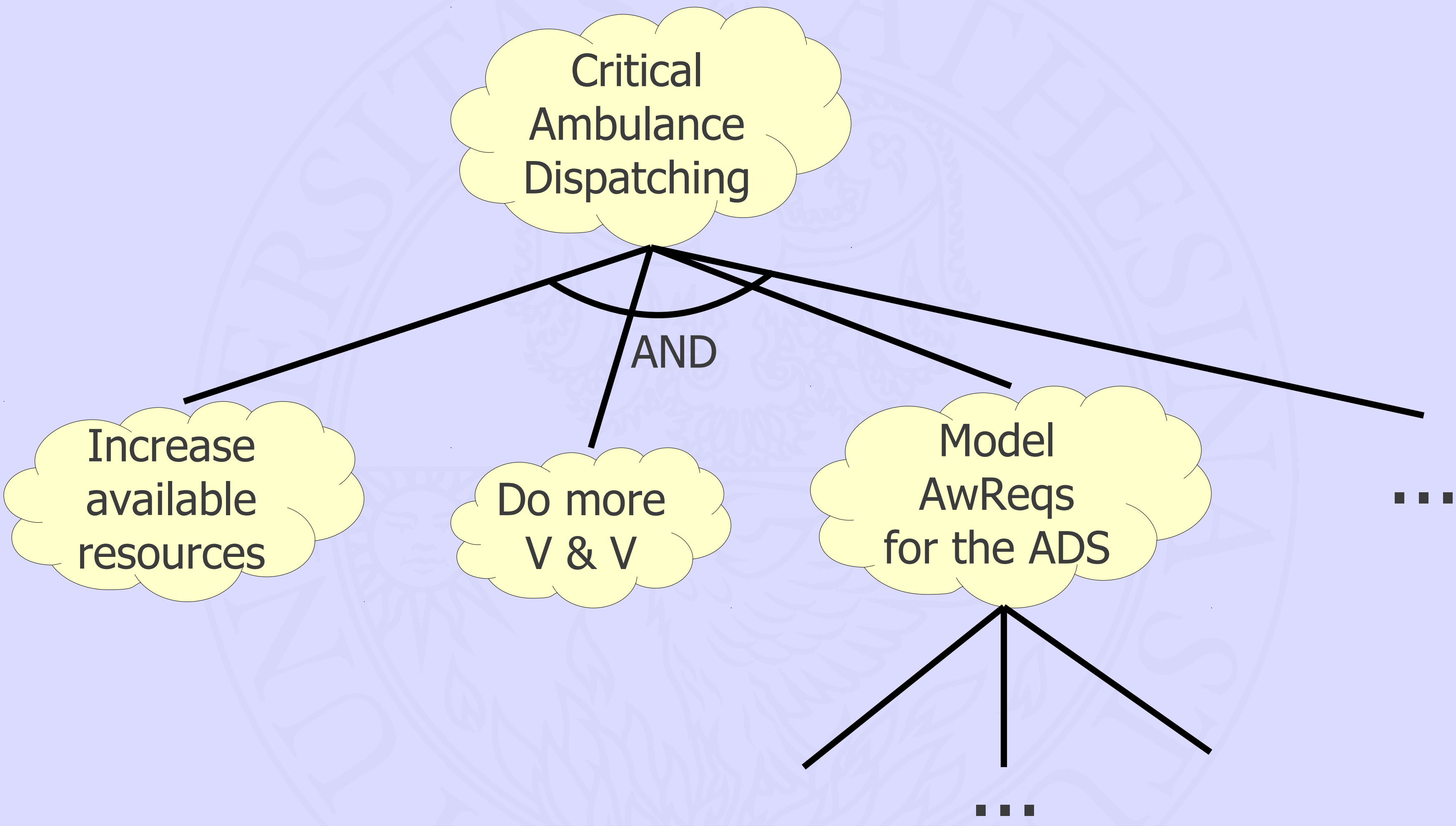
Meta-AwReqs

- AwReqs that refer to other AwReqs;
  - AR3 = SuccessRate(G-SearchCallDB, 95%, 7d);
  - AR10 = SuccessRate(AR3, 75%, 30d).
- (Examples of) motivation:
  - Gradually enforcing compensations;
  - Avoid too many compensations.



cc BY SA

Elicitation



From a talk by John Mylopoulos at University of Technology Sydney, Australia – Sep 28th, 2010

CC BY SA

Elicitation

- After the elicitation of basic requirements;
- Some sources for AwReqs:
  - Critical or risky goals (high dependency, regulations, SLAs, safety);
  - NFRs (softgoals + quality constraints);
  - Preferable solutions in variation points (*Update automatically* should succeed 100 times more);

→ AwReqs can operationalize contribution links!

  - Trade-offs: R should fail between 5 and 10 times (failure lower bound).

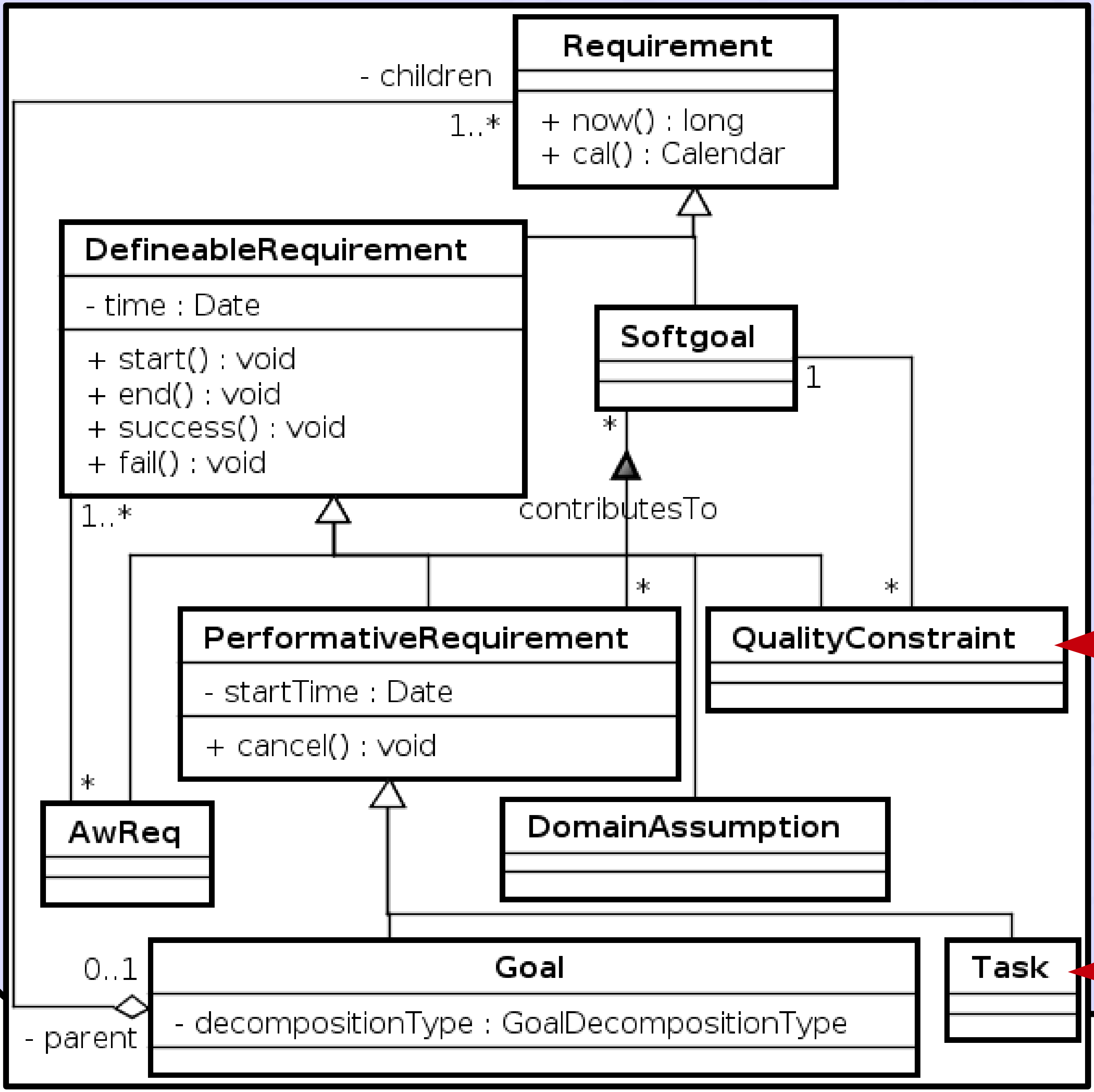
cc BY SA

Formalization

- Can use any language that:
  - Treats requirements as 1^{st} class citizens;
  - Talks about their state in different time points.
- For our experiments, we chose OCL_{TM} [Robinson, 2007];
- General approach:
  - Requirements represented as UML classes;
  - Execution at runtime = instance;
  - AwReqs written as OCL_{TM} constraints.

cc BY SA

Class model for requirements



Q-Amb10min extends this

T-InputInfo extends this

etc...

CC BY SA

Formalization examples

```
context T-InputInfo
 inv AR1: never(self.oclInState(Failed))

context G-SearchCallDB
 def: all = G-SearchCallDB.allInstances()
 def: week = all->select(g | new Date().difference(g.time,
DAYS) <= 7)
 def: success = week->select(d | d.oclInState(Succeeded))
 inv AR3: always(success->size() / week->size() >=
0.95)
```


cc BY SA

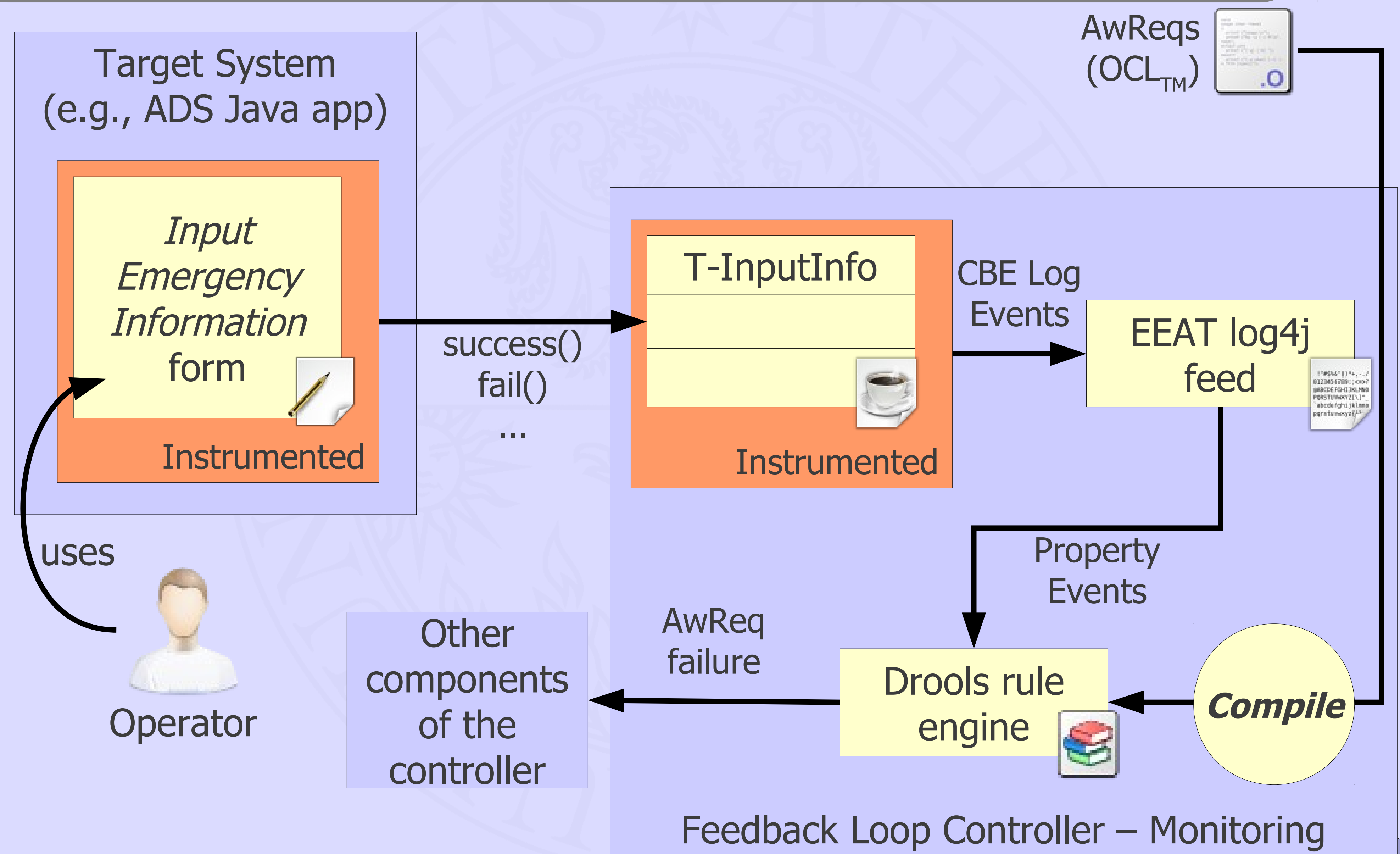
Experiment

- In a feedback loop, AwReqs relate directly to monitoring of requirements;
- There are some proposals for requirements monitoring, we adopted EEAT (ReqMon) [Robinson, 2006];
- OCL_{TM} constraints are adapted to EEAT:

```
context T-InputInfo
 def: start : LTL::OclMessage = receivedMessage('start')
 def: end : LTL::OclMessage = receivedMessage('end')
 def: fail : LTL::OclMessage = receivedMessage('fail')
 inv AR1: between(start <> null, end <> null, never(fail <> null))
```


CC BY SA

Experiment – How it works



Experiment results

- AwReqs can be monitored;
- AwReqs can provide value for a feedback loop controller that implements adaptivity;
- Monitoring has little impact on the performance of the target system.

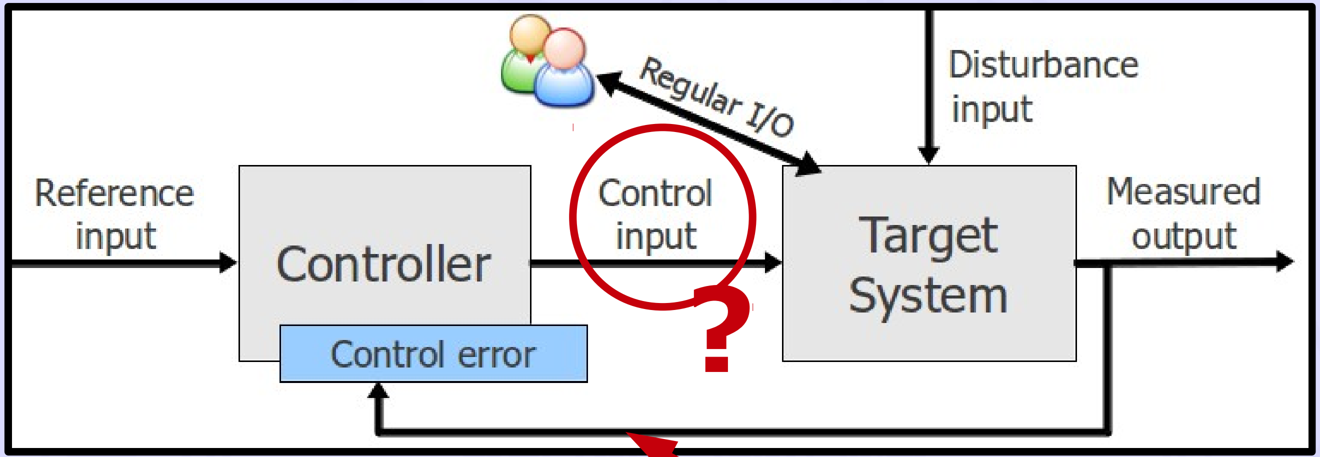
cc BY SA

Conclusions – Our Contributions

- The definition of a new class of requirements that impose constraints on the run-time behavior of other requirements;
- Linguistic constructs for expressing and formalizing such requirements;
- Fragments of a prototype implementation founded on an existing requirements monitoring framework.

cc BY SA

Current work – System Identification



Operationalized by AwReqs

- Controlled parameters x Indicators:
  - NAA = Number of ambulances available;
  - S1 = Success Rate of *Locate Available Ambulances*;
  - Δ(S1 / NAA) > 0
- Submitted to ER 2011 (Under review)

cc BY SA

Future work

- Complete MAPE loop:
  - Diagnosing, compensation, reconciliation;
  - Prototype framework;
  - CASE tools;
- For AwReqs:
  - Integration of domain models;
  - Addition of uncertainty factors (relaxed reqs.).

CC BY SA

Thank You! Questions?

Acknowledgment:

The research reported in this presentation was partially funded by the ERC advanced grant 267856 "Lucretius: Foundations for Software Evolution", unfolding during the period of April 2011 - March 2016.

This slide is sponsored by...

19th IEEE International

REQUIREMENTS ENGINEERING CONFERENCE

August 29th - September 2nd 2011 - Trento, Italy

AwReqs formalization – examples (2)

```
context T-InformDriver
 def: related : Set = T-UpdAtSite.allInstances()->select(t |
  t.arguments('callID') = self.arguments('callID'))
 inv AR8: eventually(related->size() == 1) and
  always(related->forAll(t | t.oclInState(Succeeded) and
  t.time.difference(self.time, MINUTES) <= 10))

context T-MarkUnique
 inv AR9: eventually(not self.oclInState(Undecided)) and
  never(self.time.difference(self.startTime, MINUTES) > 5)
```

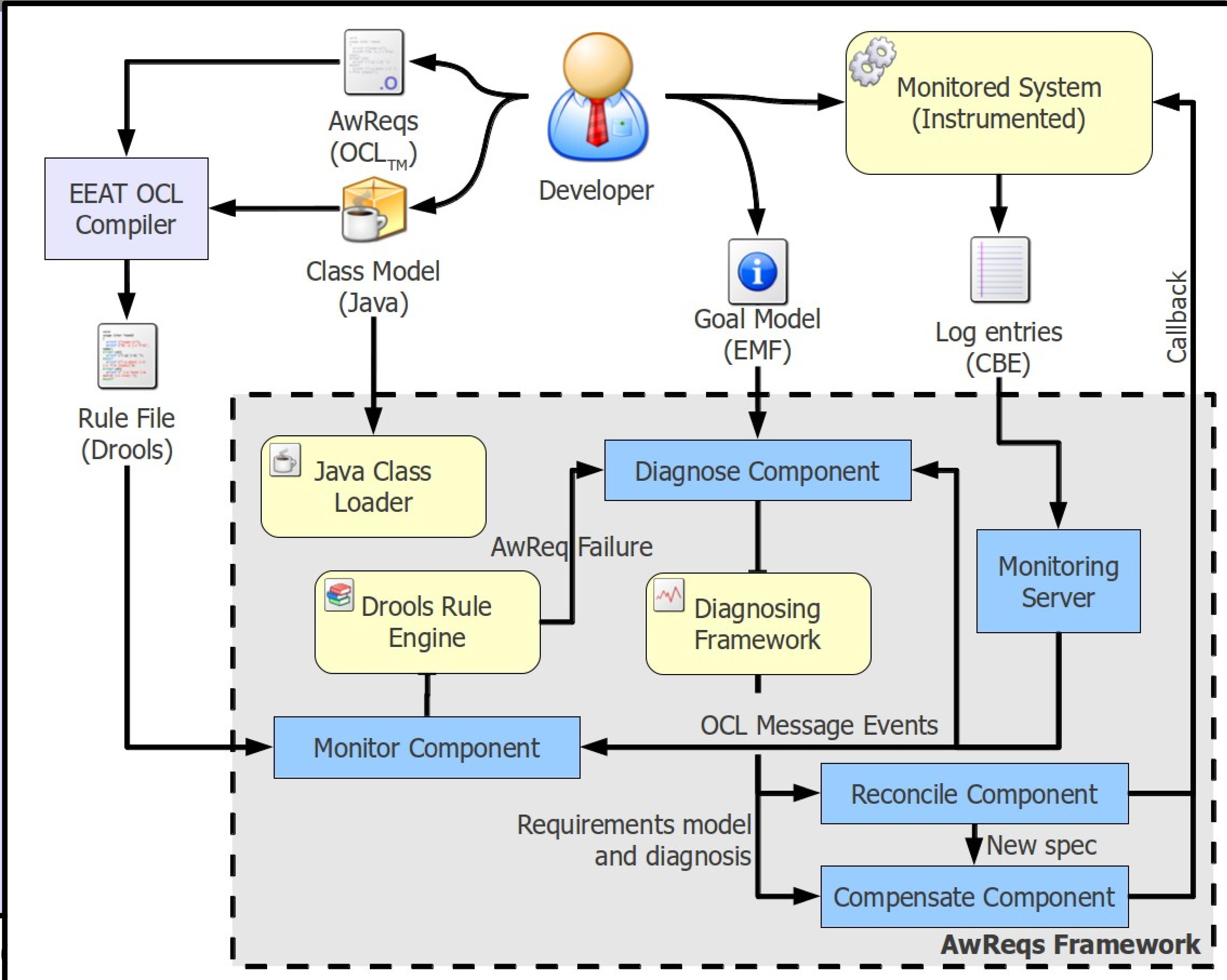

CC BY SA

Related work

- Goal-oriented, simpler approach;
  - No complex temporal constructs;
  - More suitable for requirements elicitation activities;
- Can be applied iteratively;
- Feedback loop-oriented;
  - We accept the fact that any of the requirements may eventually fail;
- Missing uncertainty factors;
- Not a full solution for adaptive systems yet.

cc BY SA

Discussion – Full feedback loop



References

- J. Andersson, R. de Lemos, S. Malek, and D. Weyns. Modeling Dimensions of Self-Adaptive Software Systems, volume 5525/2009, pages 27–47, 2009.
- Y. Brun, G. Marzo Serugendo, C. Gacek, H. Giese, H. Kienle, M. Litoiu, H. Müller, M. Pezzè, and M. Shaw. Engineering Self-Adaptive Systems through Feedback Loops, volume 5525/2009, pages 48–70, 2009.
- J. Doyle, B. Francis, and A. Tannenbaum. Feedback Control Theory. McMillan Publishing, 1990.
- M. S. Feather, S. Fickas, A. Van Lamsweerde, and C. Ponsard, "Reconciling System Requirements and Runtime Behavior," IWSSD '98: Proceedings of the 9th international workshop on Software specification and design, Washington, DC, USA: 1998, p. 50.
- J. L. Hellerstein, Y. Diao, S. Parekh, and D. M. Tilbury. Feedback Control of Computing Systems. John Wiley & Sons, 2004.
- I. J. Jureta, J. Mylopoulos, and S. Faulkner. Revisiting the Core Ontology and Problem in Requirements Engineering. In RE '08: 16th IEEE International Requirements Engineering Conference, pages 71–80, Barcelona, Spain, 2008. IEEE.
- J. O. Kephart and D. M. Chess. The vision of autonomic computing. Computer, 36(1):41–50, 2003.
- W. N. Robinson. A requirements monitoring framework for enterprise systems. Requirements Engineering, 11(1):17–41, Mar. 2006.
- W. N. Robinson. Extended OCL for Goal Monitoring. In Ocl4All '07: Proceedings of the 7th International Workshop on Ocl4All: Modelling Systems with OCL, 2007.

CC BY SA

Università degli Studi di Trento
Facoltà di Scienze Matematiche, Fisiche e Naturali
Dipartimento di Ingegneria e Scienza dell'Informazione

Awareness Requirements for Adaptive Systems

Vítor E. Silva Souza, Alexei Lapouchnian[1], William N. Robinson[2], John Mylopoulos

vitorsouza@disi.unitn.it

[1] University of Toronto, Canada / [2] Georgia State University, USA

Photo is Copyright © 2008 Andrea Caranti (http://www.flickr.com/photos/11102419@N00/1476590956/). Used with author explicit permission.